The Way We See the Problem Is the Problem

Jim York
FoxHedge Ltd

Having helped organizations implement Lean processes over the past eight years, I have had the opportunity to see a similar theme played out many times. Organizations unwilling to alter their way of seeing their problem flounder in their transformation to a Lean enterprise and ultimately regress to their former processes. Resistance to changing their way of seeing the problem is the biggest impediment for organizations trying to achieve success with Lean.

Perhaps the biggest challenge is how organizations think about value. From a Lean perspective, organizations create value when they deliver to customers what they want, when they want it, and at a price they're willing to pay [1].

For almost 100 years, the American industry's perception of value has been colored by the teachings of scientific management made popular by Frederick Winslow Taylor in his 1911 essay, "The Principles of Scientific Management." In it, Taylor states that *the most prominent single element in modern scientific management is the task idea* [2]. Taylor encourages organizations to break complex operations down into their elemental tasks, optimize the way in which each task is performed, and provide detailed instructions and plans for the execution of the task. The goals of being faster, better, and cheaper are applied at this task level. According to Taylor, the solution to maximizing productivity is discovering or developing the *one best method* to perform each task [3].

Organizations that implement scientific management also tend to measure productivity at this discrete task level. Key metrics focus on task speed, output, and cost. A problem occurs when an organization correlates these measures of task productivity with value creation.

The Perils of Local Optimization

When the task is made king, all things serve the task. A task focus is inherently myopic. It leads management to create organizational structures best suited to accomplishing the task. Efficiency experts establish the best way to execute the task through a formal process of definition, measurement, analysis, and optimization. Quality experts implement controls to ensure that the process stays within the defined tolerances. Workers are trained to complete the tasks and management evaluates their performance based on how well they comply with the process.

At its root, Lean challenges this fundamental notion of local optimization. Lean measures of success are related to cycle time, financial return, and customer satisfaction [4]. None of these Lean metrics are collected at the task level. Rather, the metrics are considered from a customer-centric perspective and applied to the entire process of getting the customer what they want. For example, cycle time measures the reliable, repeatable, and sustainable time it takes from receipt of a customer request to deployment of the solution in the customer's environment. Financial return is validation of the business case – are customers willing to pay for the results? Customer satisfaction is the ultimate proof of concept. Does the solution support the outcome the customer hoped to achieve? Evaluation of the organization's success with Lean requires foresight in identifying these customer-focused metrics and discipline in collecting and analyzing the data.

Lean in Software Development Has Its Own Challenges

Organizations that produce software are often set up to optimize at the task level. This task-level optimization results in the creation of organizational structures along specialized functions such as requirements engineering, systems architecture and design, programming, quality assurance, and operations. Work progresses in sequential phases with control gates at phase boundaries to ensure that the tasks associated with a given phase have been satisfactorily completed prior to going to the next phase. Tasks in a given phase are performed by the organizational group specializing in the principle function associated with the phase. For example, a requirements engineering group performs the tasks associated with requirements gathering and analysis phase. When a group completes their tasks and fulfills the requirements of the control gate, their work product is handed off to a group that specializes in the function performed in the next phase. Work progresses in sequential phases with control gates at phase boundaries to ensure that the tasks associated with a given phase have been satisfactorily completed.

This sequential approach to software development has been termed a *waterfall*, after a model described by Winston W. Royce in [5]. Ironically, Royce does not use the term *waterfall* to describe the model, and states that a sequential implementation of the model is *risky and invites failure* [5]. However, the model fit well with scientific management's focus on task optimization, and despite Royce's warning, many large commercial and government organizations established sequential waterfall-like development models following Royce's publication. The effects of these waterfall models are still with us today, impacting management approach, organizational structure, process, performance evaluations, and motivational systems.

Treating software development as a project can complicate matters. The Project Management Institute defines a project as a unique endeavor with a defined beginning and end [6]. For most development teams, this end occurs on acceptance and release of the application into a production environment. At this point the development team disbands and team members move on to other projects. Ongoing maintenance and enhancement of the application falls to the operations group. Who, then, is responsible for the whole life of the delivered system? Is it the development group, the operations group, or someone else?

What sequential development methods and hand-offs from development to operations fail to take into account is that customers' needs evolve and change over time. Often, customers do not really know what they need until they see a working product and then they usually need something different from what was delivered. The systems development life cycle does not end at release into production. A working product in the hands of the customer is just the beginning. After all, most of the typical application's life happens after initial deployment.

Short-term trade-offs during pre-

deployment can also have adverse impacts in the long term. For example, quality often loses the optimization battle between scope, cost, schedule, and quality. A dollar saved cutting quality upstream can cost hundreds of dollars or more later.

When looked at from a Lean perspective, the *local* optimizations just described result in decreased efficiency of the overall process serving the ultimate customer (see the "Who's Your Customer" sidebar). For example, if a customer desires a change or new feature, the change must navigate a complex change request management process. Changes disrupt the process and require considerable time to implement. In contrast, Lean sees the delivery process as a stream carrying value to the customer – the *value stream*. The object is to make value *flow* to the customer rapidly and continuously throughout the product's life. To make the transformation to a Lean enterprise, organizations must restructure work and workers to optimize flow and thus reduce cycle time.

Seeing the Value Stream

To see the value stream, organizations must step back and redefine the problem from the customers' viewpoint. Customers do not care about task efficiency. They care about results. The problem is not about task or departmental efficiency or effectiveness. Lean enterprises define the problem in terms of achieving success in the customers' eyes.

Success from the customers' perspective requires that organizations consider the entire value stream from initial concept to delivery through the entire life of a product or service, concluding with the product or service's ultimate retirement. This long-range view encourages global rather than local optimization of the delivery of value to the customer. To maintain continuity in the value stream, the team that creates the product stays with the product throughout its life. There is no handoff from a development team to an operations team: There is just one team serving the customers' needs. In this team's work queue, new features sit alongside maintenance items and are prioritized from a customer perspective with a goal of getting the highest priority items in the hands of the customer as rapidly as possible.

Agile Software Development: An Implementation of Lean Thinking

Recently, the increased popularity of Agile methodologies has led organizations to investigate and, in some cases, adopt iterative and incremental delivery. Agile software development replaces a single monolithic delivery cycle lasting several months or years with a regular cadence of brief delivery cycles lasting no more than a month. Each of these cycles produces a potentially releasable part of the overall application. The aim is to get working features to the customer sooner. Close collaboration with the customer results in delivery of the highest priority features first. The rapid delivery of tangible results permits early inspection and enables course correction when such correction can actually make a difference in getting customers what they truly need. The customer decides when to stop the regular delivery cadence based on whether the value they are realizing from the team's work justifies continued investment.

Unfortunately, many organizations fail to realize the full potential of Agile software development simply because they limit its implementation to the existing technology delivery group. This speeding up of the software development process is just fixing one part of the value stream. At the end of the day, the only thing that counts is working software in the customers' environment – by definition, the end of the stream. Combining Lean with Agile encourages a holistic view, with the customer truly becoming part of a delivery process that spans the full life of the application. Effective Agile teams already know this.

Changing Viewpoints

Successfully transitioning to a Lean enterprise requires that organizations complete the following:

- Identify your ultimate customer.
- Move from a task-centric to a customer-focused perspective.
- Establish goals for cycle time, financial results, and customer satisfaction and measure progress against these goals.
- See the whole value stream.
- Organize around flow.
- Involve the customer.

Lean is not just about doing things differently. Organizations must change the way they see the problem. The first step in this change initiative is to change yourself.◆

Who Is Your Customer?

A few years ago I was conducting a Lean-Agile overview session for a group of workers in a software shop that builds middleware for cable TV set-top boxes. During a discussion about flow of value to the customer, I posed the question, "Who is your ultimate customer?" After a few blank stares and quizzical looks, people began to call out: "My boss?" No. "The cable company?" No. "The developers that use our APIs". No. Finally, a programmer in the first row ventured: "Some dude." I asked him to explain. He went on, "Some dude sitting on a couch watching TV." Right.

References

1. Womack, James P., and Jones, Daniel T. Lean Solutions. Free Press, 2005: 15.
2. Taylor, Frederick Winslow. The Principles of Scientific Management. New York, W.W. Norton & Company, 1967:39.
3. Ibid: 25.
4. Poppendieck, Mary, and Tom Poppendieck. Implementing Lean Software Development. Boston, Pearson Education, Inc. 2007: 238-241.
5. Royce, Winston W. "Managing the Development of Large Software Systems." TRW, Proceedings IEEE WESCON Aug. 1970: 2.
6. A Guide to the Project Management Body of Knowledge. Third Edition, Newtown Square: Project Management Institute, Inc., 2004.

About the Author

Jim York is a Certified Scrum Trainer and co-founder of FoxHedge Ltd. For more than 20 years as a management and IT consultant, he has led, trained, and coached hundreds of individuals, teams, and organizations in the implementation of both Lean and Agile concepts. York's workshops blend his practical experience in Scrum, Lean Software Development, eXtreme Programming, Agile project management, product management, and traditional project management. He shares his passion for Lean and Agile as a frequent presenter at conferences, users groups, public and on-site workshops, and as a business process coach.

FoxHedge Ltd
18899 Maplewood LN
Leesburg, VA 20175
Phone: (703) 771-8367
E-mail: jim.york@mac.com